ELMB Software

ELMBio General-Purpose Application for ELMB (+ Motherboard): Status & Developments

Henk Boterenbrood

NIKHEF

ELMBio v3.3: Overview (1)

- General-purpose *CANopen* application firmware
  - For use with 'Motherboard' version 3 (and older)
  - Binary, source code and documentation on the web
- I/O
  - <= 64 analog inputs (16-bits)
  - analog outputs (*dummy: no hardware supported*)
  - 8 to 16 digital inputs
  - 8 to 16 digital outputs

ELMBio v3.3: Overview (2)

- *CANopen*
  - I/O-data communication with *PDO*s (configurable: SYNC, timer-triggered, *Change-of-State*)
  - Object Dictionary reading and writing (*SDO*s)
  - *Lifeguarding*, *Nodeguarding* (compile option: *Heartbeat*)
- Other
  - Configurability (thru Object Dictionary)
  - Configuration + parameter storage in non-volatile memory (EEPROM)
  - Different application configurations possible using compile options, e.g. for on-chip ADC, more rad-tolerant variable storage, heartbeat, Motherboard version…

ELMBio v3.3: Analogue Inputs

- Hardware choice (compile option)
  - On-board ADC (64 chan, 16-bit) or on-processor ADC (8 chan, 10-bit, Motherboard J7 connector)
- Configurable
  - Number of channels actually used
  - ADC settings: gain range, conversion wordrate
  - *CANopen* read-out mode: *PDO* triggered by *SYNC* or *RTR* or *Timer* (in secs, up to 65), 1 channel/CAN-message
  - Enable recalibration/configuration refresh before each channel scan *
- Configuration (ADC, *CANopen*) stored on-board

** in red: new features*

ELMBio v3.3: Digital Inputs

- Configurable
  - 8 to 16 inputs (8 lines can be defined as either in- or output) *
  - *CANopen* read-out mode: *PDO* triggered by *Change-of-State* (polled), *SYNC* or *RTR* or *Timer* (in secs, up to 65), all inputs transferred in one 2-byte CAN-message
  - *Change-of-State* polling can be enabled/disabled per input *
  - Debouncing
- Configuration stored on-board

** in red: new features*

ELMBio v3.3: Digital Outputs

- Configurable
  - 8 to 16 outputs (8 lines can be defined as either in- or output) *
  - Initialized to all either 0 or 1 at *hard* reset (power-up, watchdog, reset-button), but unchanged on *soft* reset (*NMT Reset-Node*) *
- Configuration stored on-board
- Set all 16 outputs by a 2-byte CAN-message (*PDO*)

** in red: new features*

ELMBio v3.3: Radiation Tolerance

- FLASH memory (program code) appears safe
- EEPROM storage (configuration) appears safe
- RAM is vulnerable, so must minimize its use
  - All* global variables (configuration) are reread from EEPROM before every use (compile option)
  - Always need some RAM, so need watchdog (=2nd microcontroller)
- Registers are vulnerable too
  - Microcontroller I/O-Ports and Data-Direction Regs: refresh all*
  - CAN-controller: general refresh only at *Lifeguarding* time-out, other problems to be detected by host and resolved e.g. by soft reset
  - ADC: optional periodic recalibration/configuration refresh *

**** in red: new features***

ELMBio v3.3: Other…

- There are several compile options…
  - Which ones are set can be read from the Object Dictionary (bit mask) *

** in red: new features*

Under development: *ELMBio v3.4*

- Support for 'real' analog outputs: DAC (hardware finalised)
  - Four modules of 16 channels each: 64 channels (12-bit)
  - Can be connected (in a chain) directly to Motherboard 20-pins J8 connector
  - Single channel setting by one CAN-message (*PDO*) containing analog output channel number + data (raw 12-bit DAC value)
  - Vulnerability to radiation and what to do against it: remains an open question…
- Other ?…

Additional New Features ?... (1)

- for ADC
  - Channel upper and lower limit (window) check (thru polling)
    - Scanning period
    - Limits per analog input channel can be set, stored in EEPROM
    - (Async) data transfer only in case of crossing a limit
  - Delta change (thru polling)
    - Scanning period
    - Delta per analog input channel can be set, stored in EEPROM
    - (Async) data transfer only in case of a delta increase/decrease
  - Increase timer-triggered read-out to intervals > 65 s
    (actually limitation by *CANopen*: 16-bit value in ms)

Additional New Features ?... (2)

- for DAC
  - Set a 4-20 mA current output, e.g. using a μA value in the CAN-message (ELMB does the conversion to a DAC value)
  - Ramping function, e.g. to control a HV-supply
    - End value and ramping speed (or time) to be set

Additional New Features ?... (3)

- Diagnostic *PDO*
  - CAN-controller is able to automatically reply to an *RTR* *(Remote Frame)*
  - The ELMB application checkpoints its status continuously in the CAN-controller's message buffer
  - Enables a host system to monitor ELMB microcontroller state without disturbing it (except for some extra overhead)
  - Can serve to determine ELMB state when micro somehow stuck…
- More features to improve rad-tolerance ?
  - e.g improved CAN-controller register refresh mechanism
- *…it's up to the users, more or less…*